#16
10-25-03
[illegible]

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of:	Confirmation No.: 8755
Sharon S. Liu, et al.	Group Art Unit No.: 3621
Serial No.: 09/483,723	Examiner: James A. Reagan
Filed: January 14, 2000	

For: OBJECT ORIENTED MECHANISM FOR DYNAMICALLY CONSTRUCTING CUSTOMIZED IMPLEMENTATIONS TO ENFORCE RESTRICTIONS

MS Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

This Appeal Brief is submitted in support of the Notice of Appeal filed on September 26, 2003.

I. REAL PARTY IN INTEREST

Sun Microsystems, Inc. is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

Appellant is unaware of any related appeals or interferences.

III. STATUS OF CLAIMS

Claims 11-23, 34-46, and 57-105 are pending in this application, have been finally rejected, and are the subject of this appeal.

IV. STATUS OF AMENDMENTS

No amendments were filed after the Final Office Action mailed on June 26, 2003.

V. SUMMARY OF THE INVENTION

For a number of years, the U.S. Department of Commerce has regulated, and at times, prohibited the exportation of computer programs or applications that implement data encryption algorithms. Currently, computer programs cannot, as a general rule, be exported if they implement encryption algorithms having cryptographic key sizes exceeding a certain number of bits (the specific allowable key size is algorithm-specific).

These regulations apply not only to programs which directly implement encryption algorithms, but also to programs which interface with programs that directly implement encryption algorithms. These programs include "framework" programs that provide infrastructure for facilitating interaction between various programs. The framework itself may not implement any encryption algorithm, but it may allow one or more programs that do implement encryption algorithms to interface with or "plug in" to the framework. If a framework allows an encryption mechanism to be "plugged in" to the framework, the framework itself will be subject to export regulations. This means that in order to be exportable, the framework needs to ensure that all export regulations are adhered to regardless of the encryption implementation that is plugged in to the framework. In order to do this, the framework needs to have some mechanism for enforcing the necessary restrictions on the encryption implementations.

The invention recited in claims 11-23, 34-46, and 57-105 provides an object-oriented mechanism for dynamically constructing customized implementations to enforce restrictions on services. For purposes of the present invention, a service is defined broadly to encompass any functionality requested by and provided to an application, including but not limited to encryption/decryption functionality. In one embodiment, the invention is implemented in a system comprising an application, a general implementation of a particular service, and a framework.

The framework receives from the application a request for an implementation of a particular service, such as an implementation of a particular encryption algorithm. In response, the framework determines what restrictions, if any, need to be imposed on the requested implementation. Once the restrictions are determined, the framework dynamically constructs the requested implementation. The requested implementation is constructed such that it incorporates the general (i.e. unrestricted) implementation of the service, the restrictions, and enforcement logic for enforcing the restrictions on the general implementation. Since the requested implementation is constructed specifically for the application, it is customized for the application. Thus, the implementation is referred to as the customized implementation (Specification, pages 1-3).

In dynamically constructing the customized implementation, the framework instantiates a wrapper instance. The wrapper instance comprises enforcement logic for enforcing the restrictions. The framework then encapsulates the restrictions and an instance of the general implementation within the wrapper instance. **In encapsulating**

the general implementation instance within the wrapper instance, the framework maps the methods of the wrapper instance to corresponding methods of the general implementation instance. This mapping enables calls to the methods of the wrapper instance to be routed to the proper methods of the general implementation instance. This is as it should be, since the implementations for these methods are provided by the general implementation instance. In contrast to the methods of the general implementation instance, however, the wrapper instance will ensure that the restrictions are enforced.

Once the encapsulation process is complete, the framework provides the newly constructed wrapper instance to the application that requested the customized implementation. The wrapper instance is provided as the customized implementation requested by the application. Due to the mapping of the methods of the wrapper instance to the proper methods of the general implementation instance, the application may invoke the methods of the wrapper instance in the same manner as the application would have invoked the methods of the general implementation instance. Consequently, the application does not need to be modified to handle the wrapper instance differently than the general implementation instance (Specification, pages 14-15).

Since the customized implementation incorporates the restrictions and enforcement logic for enforcing the restrictions, it is not necessary for the application to further interact with the framework. The customized implementation itself will provide the services, and will guarantee that the restrictions are enforced. By dynamically constructing customized implementations in this manner, the framework ensures that the necessary restrictions are enforced on the services provided to the application (Specification, page 3).

VI. ISSUES

The sole issue is whether Claims 11-23, 34-46, and 57-105 are unpatentable under 35 U.S.C. § 103(a) over U.S. Patent Application Publication No. US 2002/0112171 A1 by Ginter, et al. ("Ginter") in view of U.S. Patent No. 6,389,534 B1 issued to Elgamal, et al. ("Elgamal") in view of Pressman, "Software Engineering: A Practitioner's Approach" ("Pressman").

VII. GROUPING OF CLAIMS

Claims 11-23, 34-46, and 57-105 fall or stand together.

VIII. ARGUMENTS

A. Introduction

In the Final Office Action dated June 26, 2003, the Examiner rejected Claims 11-23, 34-46, and 57-105 under 35 U.S.C. § 103(a) as being unpatentable over Ginter in view of Elgamal and Pressman. Based upon the reasons given below, and contrary to the Examiner's contention, it is respectfully submitted that Claims 11-23, 34-46, and 57-105 are patentable over Ginter, Elgamal, and Pressman for at least the reasons provided hereinafter.

It is well founded that to establish a *prima facie* case of obviousness under 35 U.S.C. § 103(a), the references cited and relied upon must teach or suggest all the claim limitations. In addition, a sufficient factual basis to support the obviousness rejection must be proffered. *In re Freed*, 165 USPQ 570 (CCPA 1970); *In re Warner*, 154 USPQ

173 (CCPA 1967); *In re Lunsford*, 148 USPQ 721 (CCPA 1966). With respect to the present application, it is respectfully submitted that Ginter, Elgamal, and Pressman, taken alone or in combination, do not in any way teach or suggest all the limitations of Claims 11-23, 34-46, and 57-105. It is further submitted that a sufficient factual basis has not been proffered in the Final Office Action mailed June 26, 2003 to support the rejection of Claims 11-23, 34-46, and 57-105 under 35 U.S.C. § 103(a) as being unpatentable over Ginter in view of Elgamal and Pressman.

B. The Limitations of Claims 11-23, 34-46, and 57-105 Are Not in Any Way Taught or Suggested by Ginter, Elgamal, and Pressman

Claim 11 addresses the problem of how to ensure that restrictions are enforced on an implementation of a service that is plugged in to a framework. According to the approach recited in Claim 11, the framework receives, from an application, a request for a customized implementation of a particular service. The framework instantiates an implementation class that provides an implementation for the particular service. The instantiation of the implementation class gives rise to an implementation instance, which comprises one or more invocable methods. The framework determines a set of restrictions to be imposed on the customized implementation. The framework instantiates a wrapper class that comprises enforcement logic for enforcing the restrictions. The instantiation of the wrapper class gives rise to a wrapper instance, which also comprises one or more invocable methods. The framework encapsulates the implementation instance and the restrictions within the wrapper instance. In doing so, the framework maps the invocable methods of the wrapper instance to the invocable methods of the implementation instance. The framework provides the wrapper instance to the application as the customized implementation. Because the wrapper instance comprises logic for enforcing restrictions, and because the restrictions are encapsulated within the wrapper instance along with the implementation instance, the customized implementation will provide the particular service, and restrictions on that particular service will be enforced.

Summary of Examiner's Contentions

In rejecting Claim 11, the Examiner contends that Elgamal discloses several, but not all, of the above features. The Examiner admits that Elgamal does **not** disclose or suggest "providing said wrapper instance to the application as said customized implementation" as required by Claim 11. Recognizing this deficiency of Elgamal, the Examiner contends that Ginter discloses "providing said wrapper instance to the application as said customized implementation." Oddly, the Examiner also expressly relies upon Pressman to reject Claim 11, although the Examiner does not cite any particular limitation of Claim 11 that Pressman is relied upon to disclose or suggest.

The merits of the Examiner's contentions will be discussed in detail below, but one initial point to note is that the Examiner completely ignores the last limitations of Claim 11, which recite "wherein said wrapper instance comprises one or more invocable methods, wherein said implementation instance comprises one or more invocable methods, and wherein encapsulating comprises: mapping the one or more invocable methods of said wrapper instance to the one or more invocable methods of said implementation instance." The Examiner does not even allege that either Ginter, Elgamal, or Pressman, alone or in combination, in any way disclose or suggest these limitations. Thus, on this basis alone, the Examiner has failed to make out a *prima facie* case of obviousness.

To further discuss the merits of the Examiner's contentions, provided below is a discussion of each of the references relied upon by the Examiner. Those references, taken alone or in combination, fail to disclose or suggest all of the limitations of Claim 11.

Elgamal

Elgamal discloses an application program that calls a service module, which calls a policy filter that has been configured by a policy filter initialization module. Based on the determination of the policy filter, the service module returns to the application either an error (if the operation is not allowed) or a result of the operation requested by the application program (see col. 5, line 41-42, and also col. 6, lines 42-43).

As discussed above, an initial point to note regarding Elgamal is that Elgamal does not disclose or suggest "mapping the one or more invocable methods of said wrapper instance to the one or more invocable methods of said implementation instance" as required by Claim 11.

Another point to note regarding Elgamal is that Elgamal does not disclose or suggest "providing said wrapper instance to the application as said customized implementation" as required by Claim 11. Instead, Elgamal discloses the returning of a <u>result of an operation</u> to an application. Because Elgamal returns the result rather than a <u>customized implementation</u>, the application is required to make a request to a service module each time that the application needs to perform a cryptographic operation. Such repeated requests create a performance bottleneck, which the method of Claim 11 makes it possible to avoid.

In the Final Office Action, the Examiner acknowledges this deficiency of Elgamal. Thus, even the Examiner admits that Elgamal, taken individual, fails to disclose or suggest the method of Claim 11.

Ginter

Ginter discloses a content container object that contains information content and a permissions record. The information content may be, for example, text, sound, video, or computer software. The permissions record specifies rights associated with the content container object such as, for example, who can open the container object, who can use the container object's contents, and who can distribute the container object. An end user cannot use or access the container object's contents unless the permissions record has been delivered to the end user.

As discussed above, an initial point to note regarding Ginter is that Ginter does not disclose or suggest "mapping the one or more invocable methods of said wrapper instance to the one or more invocable methods of said implementation instance" as required by Claim 11.

Specifically, it should be noted that information content such as text, sound, and video <u>does not</u> comprise any invocable methods. Therefore, when the information content is text, sound, or video, there are <u>no</u> invocable methods of an implementation instance to which invocable methods of a wrapper instance could be mapped.

Additionally, the Examiner has failed to show any disclosure or suggestion in Ginter that computer software information content comprises invocable methods.

Even if the Examiner had shown that computer software information content comprised invocable methods, the Examiner still has failed to show that Ginter discloses or suggests that methods of the content container are <u>mapped</u> to methods of computer software information content contained therein. Thus, the Examiner has failed to show that Ginter discloses or suggests mapping one or more invocable methods of a wrapper instance to one or more invocable methods of an implementation instance, as is required by Claim 11.

For at least these reasons, Ginter taken alone, fails to disclose or suggest all of the limitations of Claim 11.

Pressman

Pressman discloses general principles of object-oriented software engineering. As discussed above, it is not clear exactly which aspect of Claim 11 the Examiner attempts to show by citing Pressman. In any case, the Examiner has not shown or even alleged that Pressman discloses or suggests mapping one or more invocable methods of a wrapper instance to one or more invocable methods of an implementation instance as required by Claim 11.

For at least these reasons, Pressman taken alone, fails to disclose or suggest all of the limitations of Claim 11.

Based on the foregoing, it is therefore respectfully submitted that the Examiner has failed to show that Ginter, Elgamal, and Pressman in any way disclose or suggest at least one of the limitations of Claim 11, namely, the limitation of "mapping the one or more invocable methods of said wrapper instance to the one or more invocable methods of said implementation instance." Since the Examiner has failed to show that Ginter, Elgamal, or Pressman, taken alone or in combination, disclose or suggest all of the limitations required by Claim 11, it is respectfully submitted that the Examiner has erred in rejecting Claim 11, and that Claim 11 is patentable over Ginter, Elgamal, and Pressman.

Claims 12-23, 70-72, and 79-87 depend from Claim 11 and include all of the limitations of Claim 11. It is therefore respectfully submitted that Claims 12-23, 70-72, and 79-87 are patentable over Ginter, Elgamal, and Pressman for at least the reasons set forth herein with respect to Claim 11.

Claims 34-46, 73-75, and 88-96 contain limitations similar to Claims 11-23, 70-72, and 79-87, except in the context of a framework. It is therefore respectfully submitted that Claims 11-23, 70-72, and 79-87 are patentable over Ginter, Elgamal, and Pressman for at least the reasons set forth herein with respect to Claims 11-23, 70-72, and 79-87.

Claims 57-69, 76-78, and 97-105 contain limitations similar to Claims 11-23, 70-72, and 79-87, except in the context of a computer-readable medium. It is therefore respectfully submitted that Claims 57-69, 76-78, and 97-105 are patentable over Ginter, Elgamal, and Pressman for at least the reasons set forth herein with respect to Claims 11-23, 70-72, and 79-87.

For at least these reasons, it is respectfully submitted that Claims 11-23, 34-46, and 57-105 are not in any way taught or suggested by Ginter, Elgamal, or Pressman, alone or in combination, and are therefore patentable over Ginter, Elgamal, and Pressman.

C. Claims 11-23, 34-46, and 57-105 Are Not Rendered Obvious by Ginter, Elgamal, and Pressman Because the Examiner Has Not Provided an Adequate Motivation to Combine Ginter and Elgamal

In *In re Dembiczak*, 50 USPQ2d 1614, 1617 (1999), the Court of Appeals for the Federal Circuit (CAFC) warned against the use of hindsight in an obviousness analysis:

> Measuring a claimed invention against the standard established by section 103 requires the oft-difficult but critical step of casting the mind back to the time of invention, to consider the thinking of one of ordinary skill in the art, guided only by the prior art references and the then-accepted wisdom in the field....Our case law makes clear that the best defense against the subtle but powerful attraction of a hindsight-based obviousness analysis is **rigorous application of the requirement for a showing of the teaching or motivation to combine prior art references**. (emphasis added)

The showing must be clear and particular, and broad conclusory statements regarding the teaching of multiple references, standing alone, are not evidence. Id. at 1617.

In *In re Lee*, 61 USPQ2d 1430, 1434 (2002), the Court of Appeals for the Federal Circuit (CAFC) held that an examiner's conclusory statements did not adequately address the issue of motivation to combine. In *Lee*, an examiner held that it would have been obvious to a person of ordinary skill in the relevant art to combine the teachings of two references to produce a claimed invention. The first reference described a television set having a menu display by which a user could adjust various picture and audio functions, but the display lacked a demonstration of how to adjust the functions. The second reference described a game's video display as having a demonstration mode that showed how to play the game, but the second reference did not mention the adjustment of picture or audio functions. Id. at 1431.

In rejecting a representative claim, which recited both a demonstration mode and demonstrating the adjustment of picture functions, the examiner stated that the combination of two references "would have been obvious to one of ordinary skill in the art since the demonstration mode is just a programmable feature which can be used in many different device[s] for providing automatic introduction by adding the proper programming software" and that "another motivation would be that the automatic demonstration mode is user friendly and it functions as a tutorial." Id. at 1432.

In vacating the rejection, the CAFC held that the examiner did not adequately support the selection and combination of the references. The CAFC explained:

> The examiner's conclusory statements that 'the demonstration mode is just a programmable feature which can be used in many different device[s] for providing automatic introduction by adding the proper programming software' and that 'another motivation would be that the automatic demonstration mode is user friendly and it functions as a tutorial' **do not adequately address the issue of motivation to combine. This factual question of motivation is material to patentability, and could not be resolved on subjective belief and unknown authority**. (emphasis added)

In the present case, the Examiner similarly has failed to adequately address the issue of motivation to combine the references. In the present case, the Examiner attempts to establish a motivation to combine Ginter with Elgamal based on the conclusory statement that the combination would allow an application to "travel" or be whole without having to keep referring back to a framework for restrictions. The Examiner's belief that one would be motivated to combine the teachings of Ginter and Elgamal to allow an application to "travel" or be whole without having to keep referring back to a framework for restrictions is completely subjective, and the authority on which this belief is founded is completely unknown.

The Examiner's mere summarization of a feature disclosed by Ginter is not in and of itself a clear and particular showing why one of ordinary skill in the art would have been motivated to apply that feature to the system of Elgamal. The applications disclosed by Elgamal do not appear to travel at all, so it is not clear why one of ordinary skill in the art would have been motivated to enable those applications to travel without referring to a framework for restrictions. The Examiner has not set forth any reasons as to why one of ordinary skill in the art would have equated any aspect of the system disclosed in Elgamal with the content information, disclosed in Ginter, upon which access restrictions may be placed.

Like the motivations to combine presented by the examiner in *Lee*, the motivation to combine alleged by the Examiner in the present case is neither clear nor particular, and can only be characterized as being both broad and conclusory. The Examiner's broad, conclusory statements regarding the teachings of Ginter, Elgamal, and Pressman, standing by themselves, are not evidence. Consequently, Appellant submits that the Examiner has not met his burden in making out a *prima facie* case of obviousness in the present case. Accordingly, Appellant submits that one of ordinary skill in the art would not have been motivated to combine Ginter with Elgamal. For at least the reasons discussed above, Appellant respectfully submits to the Honorable Board that the rejection of Claims 11-23, 34-46, and 57-105 should be reversed.

IX. CONCLUSION AND PRAYER FOR RELIEF

Based on the foregoing, it is respectfully submitted the rejections of Claims 11-23, 34-46, and 57-105 under 35 U.S.C. § 103(a) lack the requisite factual and legal bases. Appellant therefore respectfully requests that the Honorable Board reverse the rejection of Claims 11-23, 34-46, and 57-105 under 35 U.S.C. § 103(a) over Ginter in view of Elgamal and Pressman.

Respectfully submitted,

HICKMAN PALERMO TRUONG &
BECKER LLP

[illegible]

Date: October 16, 2003

Christian A. Nicholes
Registration No. 50,266

1600 Willow Street
San Jose, California
95125-5106
Tel: (408) 414-1224
Fax: (408) 414-1076

CERTIFICATE OF MAILING
I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: MS Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450

on 10/16/03 by [illegible]
Jennifer Newell

CLAIMS APPENDIX

101

11. In a system comprising an application, a framework, and an implementation class which provides an implementation for a particular service, a method performed by the framework, comprising:

112

receiving a request from an application for a customized implementation of a 112 particular service;

instantiating an implementation class which provides an implementation for the particular service to give rise to an implementation instance;

determining a set of zero or more restrictions to be imposed on said customized implementation;

instantiating a wrapper class to give rise to a wrapper instance, said wrapper instance comprising enforcement logic for enforcing said restrictions;

encapsulating said implementation instance and said restrictions within said wrapper instance; and

providing said wrapper instance to the application as said customized implementation;

wherein said wrapper instance comprises one or more invocable methods, wherein said implementation instance comprises one or more invocable methods, and wherein encapsulating comprises:

mapping the one or more invocable methods of said wrapper instance to the one or more invocable methods of said implementation instance.

12. The method of claim 11, wherein instantiating the implementation class comprises:

determining whether the implementation class is authentic; and

in response to a determination that the implementation class is authentic, instantiating the implementation class to give rise to said implementation instance.

13. The method of claim 12, wherein the implementation class has a digital signature associated therewith, and wherein determining whether the implementation class is authentic comprises:

verifying said digital signature.

14. The method of claim 12, wherein the implementation class authenticates the framework prior to giving rise to said implementation instance.

15. The method of claim 11, wherein determining the set of zero or more restrictions comprises:

accessing information specifying one or more limitations; and

processing said limitations to derive said restrictions.

16. The method of claim 15, wherein the particular service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations.

17. The method of claim 16, wherein said default encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

18. The method of claim 11, wherein determining the set of zero or more restrictions comprises:

accessing information specifying one or more limitations;

determining permissions, if any, granted to the application; and

reconciling said limitations and said permissions to derive said restrictions.

19. The method of claim 18, wherein said limitations and said permissions are reconciled to derive restrictions which are least restrictive.

20. The method of claim 18, wherein the particular service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations, and a set of zero or more exempt encryption limitations which apply when one or more exemption mechanisms are implemented.

21. The method of claim 20, wherein said default encryption limitations and said exempt encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

22. The method of claim 20, wherein reconciling said limitations and said permissions comprises:

determining whether the application has been granted any permissions; and

in response to a determination that the application has not been granted any permissions, deriving said restrictions from said set of default encryption limitations.

23. The method of claim 20, wherein reconciling said limitations and said permissions comprises:

determining whether the application has been granted any permissions which require implementation of a particular exemption mechanism;

in response to a determination that the application has been granted a permission which requires implementation of a particular exemption mechanism, determining whether said exempt encryption limitations allow said particular exemption mechanism to be implemented; and

in response to a determination that said exempt encryption limitations allow said particular exemption mechanism to be implemented, deriving said restrictions from said set of exempt encryption limitations.

34. 10' In a system comprising an application and an implementation class which provides an implementation for a particular service, a framework comprising:

a mechanism for receiving a request from an 1.2 application for a customized implementation of a 1.2 particular service;

a mechanism for instantiating an implementation class which provides an implementation for the particular service to give rise to an implementation instance;

a mechanism for determining a set of zero or more restrictions to be imposed on said customized implementation;

a mechanism for instantiating a wrapper class to give rise to a wrapper instance, said wrapper instance comprising enforcement logic for enforcing said restrictions;

a mechanism for encapsulating said implementation instance and said restrictions within said wrapper instance; and

a mechanism for providing said wrapper instance to the application as said customized implementation;

wherein said wrapper instance comprises one or more invocable methods, wherein said implementation instance comprises one or more invocable methods, and wherein the mechanism for encapsulating comprises:

a mechanism for mapping the one or more invocable methods of said wrapper instance to the one or more invocable methods of said implementation instance.

35. The framework of claim 34, wherein the mechanism for instantiating the implementation class comprises:

a mechanism for determining whether the implementation class is authentic; and

a mechanism for instantiating, in response to a determination that the implementation class is authentic, the implementation class to give rise to said implementation instance.

36. The framework of claim 35, wherein the implementation class has a digital signature associated therewith, and wherein the mechanism for determining whether the implementation class is authentic comprises:

a mechanism for verifying said digital signature.

37. The framework of claim 35, wherein the implementation class authenticates the framework prior to giving rise to said implementation instance.

38. The framework of claim 34, wherein the mechanism for determining the set of zero or more restrictions comprises:

a mechanism for accessing information specifying one or more limitations; and

a mechanism for processing said limitations to derive said restrictions.

39. The framework of claim 38, wherein the particular service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations.

40. The framework of claim 39, wherein said default encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

41. The framework of claim 34, wherein the mechanism for determining the set of zero or more restrictions comprises:

a mechanism for accessing information specifying one or more limitations;

a mechanism for determining permissions, if any, granted to the application; and

a mechanism for reconciling said limitations and said permissions to derive said restrictions.

42. The framework of claim 41, wherein said limitations and said permissions are reconciled to derive restrictions which are least restrictive.

43. The framework of claim 41, wherein the particular service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations, and a set of zero or more exempt encryption limitations which apply when one or more exemption mechanisms are implemented.

44. The framework of claim 43, wherein said default encryption limitations and said exempt encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

45. The framework of claim 43, wherein the mechanism for reconciling said limitations and said permissions comprises:

a mechanism for determining whether the application has been granted any permissions; and

a mechanism for deriving, in response to a determination that the application has not been granted any permissions, said restrictions from said set of default encryption limitations.

46. The framework of claim 43, wherein the mechanism for reconciling said limitations and said permissions comprises:

a mechanism for determining whether the application has been granted any permissions which require implementation of a particular exemption mechanism;

a mechanism for determining, in response to a determination that the application has been granted a permission which requires implementation of a particular exemption mechanism, whether said exempt encryption limitations allow said particular exemption mechanism to be implemented; and

a mechanism for deriving, in response to a determination that said exempt encryption limitations allow said particular exemption mechanism to be implemented, said restrictions from said set of exempt encryption limitations.

57. In a system comprising an application and an implementation class which provides an implementation for a particular service, a computer readable medium having stored thereon instructions which, when executed by one or more processors, cause the one or more processors to implement a framework which dynamically constructs a customized implementation, said computer readable medium comprising:

instructions for causing one or more processors to receive a request from an application for a customized implementation of a particular service;

instructions for causing one or more processors to instantiate an implementation class which provides an implementation for the particular service to give rise to an implementation instance;

instructions for causing one or more processors to determine a set of zero or more restrictions to be imposed on said customized implementation;

instructions for causing one or more processors to instantiate a wrapper class to give rise to a wrapper instance, said wrapper instance comprising enforcement logic for enforcing said restrictions;

instructions for causing one or more processors to encapsulate said implementation instance and said restrictions within said wrapper instance; and

instructions for causing one or more processors to provide said wrapper instance to the application as said customized implementation;

wherein said wrapper instance comprises one or more invocable methods, wherein said implementation instance comprises one or more invocable methods, and wherein the instructions for causing one or more processors to encapsulate comprises:

instructions for causing one or more processors to map the one or more invocable methods of said wrapper instance to the one or more invocable methods of said implementation instance.

58. The computer readable medium of claim 57, wherein the instructions for causing one or more processors to instantiate the implementation class comprises:

instructions for causing one or more processors to determine whether the implementation class is authentic; and

instructions for causing one or more processors to instantiate, in response to a determination that the implementation class is authentic, the implementation class to give rise to said implementation instance.

59. The computer readable medium of claim 58, wherein the implementation class has a digital signature associated therewith, and wherein the instructions for causing one or more processors to determine whether the implementation class is authentic comprises:

instructions for causing one or more processors to verify said digital signature.

60. The computer readable medium of claim 58, wherein the implementation class authenticates the framework prior to giving rise to said implementation instance.

61. The computer readable medium of claim 57, wherein the instructions for causing one or more processors to determine the set of zero or more restrictions comprises:

instructions for causing one or more processors to access information specifying one or more limitations; and

instructions for causing one or more processors to process said limitations to derive said restrictions.

62. The computer readable medium of claim 61, wherein the particular service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations.

63. The computer readable medium of claim 62, wherein said default encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

64. The computer readable medium of claim 57, wherein the instructions for causing one or more processors to determine the set of zero or more restrictions comprises:

instructions for causing one or more processors to access information specifying one or more limitations;

instructions for causing one or more processors to determine permissions, if any, granted to the application; and

instructions for causing one or more processors to reconcile said limitations and said permissions to derive said restrictions.

65. The computer readable medium of claim 64, wherein said limitations and said permissions are reconciled to derive restrictions which are least restrictive.

66. The computer readable medium of claim 64, wherein the particular service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations, and a set of zero or more exempt encryption limitations which apply when one or more exemption mechanisms are implemented.

67. The computer readable medium of claim 66, wherein said default encryption limitations and said exempt encryption limitations are derived by merging

multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

68. The computer readable medium of claim 66, wherein the instructions for causing one or more processors to reconcile said limitations and said permissions comprises:

instructions for causing one or more processors to determine whether the application has been granted any permissions; and

instructions for causing one or more processors to derive, in response to a determination that the application has not been granted any permissions, said restrictions from said set of default encryption limitations.

69. The computer readable medium of claim 66, wherein the instructions for causing one or more processors to reconcile said limitations and said permissions comprises:

instructions for causing one or more processors to determine whether the application has been granted any permissions which require implementation of a particular exemption mechanism;

instructions for causing one or more processors to determine, in response to a determination that the application has been granted a permission which requires implementation of a particular exemption mechanism, whether said exempt encryption limitations allow said particular exemption mechanism to be implemented; and instructions for causing one or more processors to derive, in response to a determination that said exempt encryption limitations allow said particular exemption mechanism to be implemented, said restrictions from said set of exempt encryption limitations.

70. The method of claim 11, wherein determining said set of zero or more restrictions includes determining a set of zero or more restrictions that are specific to said application.

71. The method of claim 70, wherein determining said set of zero or more restrictions that are specific to said application includes determining a set of zero or more restrictions that are customized for said application.

72. The method of claim 11, wherein said set is a first set, and wherein said customized implementation is a first customized implementation, and further comprising:

receiving a request from a second application for a second customized implementation of said particular service, wherein said second customized implementation differs from said first customized implementation;

instantiating said implementation class which provides said implementation for said particular service to give rise to a second implementation instance;

determining a second set of zero or more restrictions to be imposed on said second customized implementation, wherein said second set differs from said first set;

instantiating said wrapper class to give rise to a second wrapper instance, said second wrapper instance comprising enforcement logic for enforcing said second set of zero or more restrictions;

encapsulating said second implementation instance and said second set of zero or more restrictions within said second wrapper instance; and

providing said second wrapper instance to said second application as said second customized implementation.

73. The framework of claim 34, wherein said mechanism for determining said set of zero or more restrictions includes a mechanism for determining a set of zero or more restrictions that are specific to said application.

74. The framework of claim 73, wherein said mechanism for determining said set of zero or more restrictions that are specific to said application includes a mechanism for determining a set of zero or more restrictions that are customized for said application.

75. The framework of claim 34, wherein said set is a first set, and wherein said customized implementation is a first customized implementation, and further comprising:

a mechanism for receiving a request from a second application for a second customized implementation of said particular service, wherein said second customized implementation differs from said first customized implementation;

a mechanism for instantiating said implementation class which provides said implementation for said particular service to give rise to a second implementation instance;

a mechanism for determining a second set of zero or more restrictions to be imposed on said second customized implementation, wherein said second set differs from said first set;

a mechanism for instantiating said wrapper class to give rise to a second wrapper instance, said second wrapper instance comprising enforcement logic for enforcing said second set of zero or more restrictions;

a mechanism for encapsulating said second implementation instance and said second set of zero or more restrictions within said second wrapper instance; and

a mechanism for providing said second wrapper instance to said second application as said second customized implementation.

76. The computer readable medium of claim 57, wherein said instructions for determining said set of zero or more restrictions include instructions for determining a set of zero or more restrictions that are specific to said application.

77. The computer readable medium of claim 76, wherein said instructions for determining said set of zero or more restrictions that are specific to said application include instructions for determining a set of zero or more restrictions that are customized for said application.

78. The computer readable medium of claim 57, wherein said set is a first set, and wherein said customized implementation is a first customized implementation, and further comprising:

instructions for receiving a request from a second application for a second customized implementation of said particular service, wherein said second customized implementation differs from said first customized implementation;

instructions for instantiating said implementation class which provides said implementation for said particular service to give rise to a second implementation instance;

instructions for determining a second set of zero or more restrictions to be imposed on said second customized implementation, wherein said second set differs from said first set;

instructions for instantiating said wrapper class to give rise to a second wrapper instance, said second wrapper instance comprising enforcement logic for enforcing said second set of zero or more restrictions;

instructions for encapsulating said second implementation instance and said second set of zero or more restrictions within said second wrapper instance; and

instructions for providing said second wrapper instance to said second application as said second customized implementation.

79. The method of claim 11, wherein said wrapper instance is invocable by the application without further interaction with the framework.

80. The method of claim 11, wherein the implementation class provides an unrestricted implementation for the particular service.

81. The method of claim 80, wherein the particular service is an encryption/decryption service, and wherein the unrestricted implementation provided by the implementation class is capable of using unlimited encryption/decryption parameters.

82. The method of claim 81, wherein the unrestricted implementation provided by the implementation class is capable of using encryption/decryption keys of any size.

83. The method of claim 11, wherein said enforcement logic enforces said restrictions on said implementation instance.

84. The method of claim 83, wherein said enforcement logic enforces said restrictions on said implementation instance by:

receiving a set of desired parameters from the application;

determining whether the desired parameters exceed said restrictions; and

in response to a determination that the desired parameters exceed said restrictions, preventing said implementation instance from operating.

85. The method of claim 84, wherein said enforcement logic is invoked upon initialization of said wrapper instance.

86. The method of claim 11, wherein the system further comprises an exemption mechanism class which provides an implementation for a particular exemption mechanism, and wherein said method further comprises:

instantiating the exemption mechanism class to give rise to an exemption mechanism instance; and

encapsulating said exemption mechanism instance within said wrapper instance.

87. The method of claim 86, wherein said enforcement logic is invoked upon initialization of said wrapper instance, and when invoked, said enforcement logic:

determines whether said exemption mechanism instance has been invoked; and

in response to a determination that said exemption mechanism instance has not been invoked, preventing said implementation instance from operating.

88. The framework of claim 34, wherein said wrapper instance is invocable by the application without further interaction with the framework.

89. The framework of claim 34, wherein the implementation class provides an unrestricted implementation for the particular service.

90. The framework of claim 89, wherein the particular service is an encryption/decryption service, and wherein the unrestricted implementation provided by the implementation class is capable of using unlimited encryption/decryption parameters.

91. The framework of claim 90, wherein the unrestricted implementation provided by the implementation class is capable of using encryption/decryption keys of any size.

92. The framework of claim 34, wherein said enforcement logic enforces said restrictions on said implementation instance.

93. The framework of claim 92, wherein said enforcement logic enforces said restrictions on said implementation instance by:

receiving a set of desired parameters from the application;

determining whether the desired parameters exceed said restrictions; and

in response to a determination that the desired parameters exceed said restrictions, preventing said implementation instance from operating.

94. The framework of claim 93, wherein said enforcement logic is invoked upon initialization of said wrapper instance.

95. The framework of claim 34, wherein the system further comprises an exemption mechanism class which provides an implementation for a particular exemption mechanism, and wherein said framework further comprises:

a mechanism for instantiating the exemption mechanism class to give rise to an exemption mechanism instance; and a mechanism for encapsulating said exemption mechanism instance within said wrapper instance.

96. The framework of claim 95, wherein said enforcement logic is invoked upon initialization of said wrapper instance, and when invoked, said enforcement logic:

determines whether said exemption mechanism instance has been invoked; and

in response to a determination that said exemption mechanism instance has not been invoked, preventing said implementation instance from operating.

97. The computer readable medium of claim 57, wherein said wrapper instance is invocable by the application without further interaction with the framework.

98. The computer readable medium of claim 57, wherein the implementation class provides an unrestricted implementation for the particular service.

99. The computer readable medium of claim 98, wherein the particular service is an encryption/decryption service, and wherein the unrestricted implementation provided by the implementation class is capable of using unlimited encryption/decryption parameters.

100. The computer readable medium of claim 99, wherein the unrestricted implementation provided by the implementation class is capable of using encryption/decryption keys of any size.

101. The computer readable medium of claim 57, wherein said enforcement logic enforces said restrictions on said implementation instance.

102. The computer readable medium of claim 101, wherein said enforcement logic enforces said restrictions on said implementation instance by:

receiving a set of desired parameters from the application;

determining whether the desired parameters exceed said restrictions; and

in response to a determination that the desired parameters exceed said restrictions, preventing said implementation instance from operating.

103. The computer readable medium of claim 102, wherein said enforcement logic is invoked upon initialization of said wrapper instance.

104. The computer readable medium of claim 57, wherein the system further comprises an exemption mechanism class which provides an implementation for a particular exemption mechanism, and wherein said computer readable medium further comprises:

instructions for causing one or more processors to instantiate the exemption mechanism class to give rise to an exemption mechanism instance; and

instructions for causing one or more processors to encapsulate said exemption mechanism instance within said wrapper instance.

105. The computer readable medium of claim 104, wherein said enforcement logic is invoked upon initialization of said wrapper instance, and when invoked, said enforcement logic:

determines whether said exemption mechanism instance has been invoked; and

in response to a determination that said exemption mechanism instance has not been invoked, preventing said implementation instance from operating.